⑲日本国特許庁（JP）　⑪特許出願公開

⑫公開特許公報（A）　平3−237887

⑤Int.Cl.⁵	識別記号	庁内整理番号	㊸公開　平成3年(1991)10月23日
H 04 N 7/133	Z	6957−5C	
H 03 M 7/30		6832−5J	

審査請求　未請求　請求項の数 2　（全9頁）

㊹発明の名称　ＤＣＴ処理装置

㉑特　願　平2−34310
㉒出　願　平2(1990)2月14日

㉗発明者　藤原　美貴雄　大阪府門真市大字門真1006番地　松下電器産業株式会社内
㉗発明者　峯丸　貴行　大阪府門真市大字門真1006番地　松下電器産業株式会社内
㉗発明者　高山　久　大阪府門真市大字門真1006番地　松下電器産業株式会社内
㉑出願人　松下電器産業株式会社　大阪府門真市大字門真1006番地
㉔代理人　弁理士　粟野　重孝　外1名

明　細　書

1. 発明の名称

ＤＣＴ処理装置

2. 特許請求の範囲

（1）画像信号の帯域圧縮で用いられるＤＣＴ処理において、Ｍビット長の信号をＮ×Ｎ画素の処理単位でＤＣＴ処理を行なう場合に、Ｍ＞Ｎの関係が成立する時、Ｍビット長をＬ＜Ｎを満足するＬビット長の信号に分割し、各Ｌビット長の部分積の演算をビットシリアルに加算器とＲＯＭを用いて演算を実行し、最後にそれらの演算結果を加算することにより、Ｎ回のサンプリングクロック期間で、Ｍビット長のＮ×１の一次元ＤＣＴ処理を完結することを特徴とするＤＣＴ処理装置。

（2）上記Ｍビット長のＮ×１の一次元ＤＣＴ処理装置２個とデータ列のスキャン方向を変換するデュアルポートメモリを用いることを特徴とするＭビット長のＮ×Ｎの二次元ＤＣＴ処理装置。

3. 発明の詳細な説明

産業上の利用分野

本発明は、テレビ会議システム、テレビ電話の動画像帯域圧縮でＣＣＩＴＴにより標準化作業がなされている64kビット／秒の画像コーデック処理で用いられるＤＣＴ(Discrete Cosine Transform、離散コサイン変換)処理装置に関する。

従来の技術

１画素データがＭビット長であるＭ×Ｎ画素ブロックに対して、ＤＣＴを行なう場合、フィルター処理等の場合と異なり、Ｎ画素のデータアクセス期間中に、一次元方向の処理が完結していれば良いという利点がある。この利点を活用して、ビットシリアルに演算処理をおこなう方法が、分散型演算手法として、例えば、アイ・イー・イー・イー・トランザクション・アコースティックス、スピーチ、シグナル、プロセッシング第22巻(1974年12月)第456頁から第462頁(ＩＥＥＥTrans.Acoustic.,Speech,Signal Processing vol.ASSP-22, pp.456-462,Dec.1974,"A new hardware realizat

ion of digital filters," by A.Peled and B.Liou)に発表されている。この処理手法は、Mビット長のデータに関する演算を、iビット目の演算というサブセットに着目して算出し、その結果に対して$2^{(i-1)}$の桁補正を施して加算することにより最終結果を求めるというものである。DCT処理について、この手法を適用すると、以下のようになる。今、Mビット長で負の数を2の補数で表わすN個の整数データ列$\{u(n)=\sum_{i=0}^{M-1}a_i(n)2^i, a_{M-1}(n)=[0,-1], (a_i(n)=[0,1], 0\leq i\leq M-2, 0\leq n\leq N-1)\}$に対する一次元のDCTは、式(1-1)～(1-3)と表現することが出来る。

$$\nu(k)=\alpha(k)\sum_{n=0}^{N-1}u(n)\cos\left[\frac{\pi(2n+1)k}{2N}\right], 0\leq k\leq N-1 \quad (1-1)$$

$$\alpha(0)=\left(\frac{1}{N}\right)^{1/2} \quad (1-2)$$

$$\alpha(k)=\left(\frac{2}{N}\right)^{1/2}, \quad 0\leq k\leq N-1 \quad (1-3)$$

上式(1-1)に、u(n)の指数表現を代入すると、式(1-4)のように書ける。

$$\nu(k)=\alpha(k)\sum_{n=0}^{N-1}\sum_{i=0}^{M-1}a_i(n)2^i\cos\left[\frac{\pi(2n+1)k}{2N}\right], 0\leq k\leq N-1 \quad (1-4)$$

この式で、iに関する加算でまとめると、次式のようになる。

$$\nu(k)=\sum_{i=0}^{M-1}2^i\left(\sum_{n=0}^{N-1}a_i(n)\alpha(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right]\right), 0\leq k\leq N-1 \quad (1-5)$$

式(1-5)で、大括弧( )の中のデータで、$a_i(n)$は0か1あるいは0か－1の1ビットのデータであり$\alpha(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right]$はデータu(n)の値そのものには依存しないので、Nの値が決まれば事前に準備することが可能である。故に、大括弧( )の中の演算は、$\alpha(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right]$の値をROM(Read Only Memory)等で準備しておけば、乗算を用いることなく加減算のみで実行することが出来、集積回路で実現する場合に、並列乗算器を用いる場合に比べてチップサイズを小さくすることが出来る利点を有している。さらにDCTの場合、変換核の$\cos\left[\frac{\pi(2n+1)k}{2N}\right]$がnにたいして周期($\pi$)で対称性を有することを利用すると、Nが偶数の場合、N＝2N'として式(1-5)は以下のようにあらわすことが出来る。

$$\nu(k)=\sum_{i=0}^{M-1}2^i\left\{\sum_{n=0}^{N'-1}a_i(n)\alpha(k)\cos\left[\frac{\pi(2n+1)k}{4N'}\right]+\sum_{n=0}^{N'-1}a_i(2N'-1-n)\alpha(k)\cos\left[\frac{(2(2N'-1-n)+1)k}{4N'}\right]\right\}, 0\leq k\leq 2N'-1 \quad (1-6)$$

上式の第二項のcos(・)の項を変形すると、

$$\cos\left[\frac{\pi(2(2N'-1-n)+1)k}{4N'}\right]=-\cos(\pi k)\cos\left[\frac{-(2n+1\pi k}{4N'}\right]$$

$$-\cos(\pi k)\cos\left[\frac{(2n+1)\pi k}{4N'}\right] \quad (1-7)$$

となり、K=2K', $0\leq k'\leq N'-1$の時

$$\cos\left[\frac{\pi(2(2N'-1-n)+1)k}{4N'}\right]=-\cos\left[\frac{(2n+1)\pi k}{4N'}\right] \quad (1-8)$$

同様に、K=2K'+1, $0\leq k'\leq N'-1$の時

$$\cos\left[\frac{\pi(2(2N'-1-n)+1)k}{4N'}\right]=-\cos\left[\frac{(2n+1)\pi k}{4N'}\right] \quad (1-9)$$

となる。式(1-8),(1-9)を用いて、kについて偶数項と奇数項で式(1-7)を変形すると、次式のようになる。

k=2K', $0\leq k'\leq N'-1$の時

$$\nu(2k')=\sum_{i=0}^{M-1}2^i\left\{\sum_{n=0}^{N'-1}(a_i(n)+a_i(2N'-1-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{4N'}\right]\right\}, 0\leq k'\leq N'-1 \quad (1-10)$$

k=2k'+1, $0\leq k'\leq N'-1$の時

$$\nu(2k'+1)=\sum_{i=0}^{M-1}2^i\left\{\sum_{n=0}^{N'-1}(a_i(n)-a_i(2N'-1-n))\cdot\alpha(2k'+1)\cos\left[\frac{\pi(2n+1)(2k'+1)}{4N'}\right]\right\}, 0\leq k'\leq N'-1 \quad (1-11)$$

となる。

式(1-10)と(1-11)により、DCTの変換核$\cos\left[\frac{\pi(2n+1)k}{2N}\right]$の対称性を利用すると、$\alpha(k)\cos$

$[\frac{\pi(2n+1)k}{2N}]$として準備しておくべきROMの容量は2^M 1つのkに対して2^Mワードから$2^{(M-3)}$に節約することが出来ることがわかる。しかし、$(a_1(n)+a_1(2N'-1-n))$や$(a_1(n)-a_1(2N'-1-n))$項から、キャリーおよびボロー発生があるので、iに関する加算回数は(M+1)回となる。このように、この演算方式は、大括弧[ ]の中の演算を、DCTの変換核$\alpha(k)\cos[\frac{\pi(2n+1)k}{2N}]$の値をROM化すると同時にDCTの変換核の対称性を利用してROM容量を節約することができ、演算そのものは乗算を用いることなく加減算のみで実行することが出来る。これらの特徴は、集積回路で実現する場合に、並列乗算器を用いる場合に比べてチップサイズを小さくすることが出来るという利点を有している。

発明が解決しようとする課題

しかしながら、1画素のサンプリング時間が1基本クロック期間であるとして、この1クロック期間に一回の加算処理や一回のROMアクセスが可能な同期系を想定すると、ビット長Mが、DCTの処理単位Nよりも大きい場合、そのままでは、処理が完結しないことを意味する。これは、N=16以上の場合には問題にならないが、CCITTにより標準化作業がなされている64kビット/秒の画像コーデック処理で用いられるN=8のDCTの場合には、M≦8ビットで制限されることになるため、中間処理部で十分な精度を得られないという問題点があった。本発明はかかる点に鑑み、M>Nビットの精度でNサンプリングクロックの期間でN×1の一次元のDCT処理を完結するN×NのDCT処理装置を安価に提供することを目的とする。

課題を解決するための手段

上記の問題点を解決するため、本発明のDCT処理装置は、Mビット長をL<Nを満足するLビット長に分割し、Lビット長で部分積の演算を並列的に実行し、最後にそれらの中間結果の加算を実行するという構成を備えたものである。

作用

本発明は前記した構成により、Lビット長で部分積の演算が並列に実行されると、中間和が並列に生成されるために、演算が高速に実行されることとなり、ビット長MがDCTの処理単位Nよりも大きい場合においてもNサンプリングクロックの期間で処理が完結する。

実施例

以下、本発明のDCT処理装置の一実施例を図面と共に説明する。第1図は本発明の一実施例における14ビットの画像信号入力u(j)に対する8×1の一次元のDCT処理装置のブロック図である。図において、2は14ビットの画像信号入力u(j)、3～10は14ビットのデータレジスタ、11～18は14ビットの画像信号{u(n-mod(j)8)、n=0～7}である。19～22はビットシリアル演算部であり、シフトレジスタを用いて、ビットシリアルに加算および減算を行なう。23～38はビットシリアル演算部19～22のビットシリアル演算の結果である各1ビットの信号で、39～42は1ビットの演算結果23～38を各4ビットごとにまとめたデータ線である。43～46はデータ線39～42の4ビットのデータをアドレス情報とし、ROMにより係数とデータの乗算の部分積を生成し、その値に左方シフトを施し累積加算を行なうROMと加算器による係数乗算部である。47～54は8×1のDCT処理結果の33ビットの出力信号{v(k),k=0～7}である。55～62は33ビットトライステートドライバであり、出力データの並列/直列変換を行なう。63は33ビットトライステートドライバの55～62の動作により時系列化された33ビット信号出力である。第2図は第1図のビットシリアル演算部19～22の回路構成図である。65は14ビットの画像信号u(n)、66は14ビットの画像信号u(7-n)である。67,68は上位7ビットと下位7ビットが独立な14ビットのデータロード機能付き右方シフターであり、ビットシリアル演算に必要なビット単位での処理を行なう。69,70は1ビット全加算器、71,72は1ビット全減算器である。73～76は1ビットのデータラッチで、1ビット全加算器69,70での演算で発生するキャリーおよび1ビット全減算器71,72での演算で発生するボローを保持する。77～80は各1ビットの演算結果の

信号であり、係数との乗算の部分積をＲＯＭから読み出す時のアドレス情報として用いられる。第３図は第１図のＲＯＭと加算器による係数乗算部43～46の回路構成図である。82～85は係数との乗算の部分積をＲＯＭから読み出す時のアドレス情報である各４ビットのデータである。86～89は16ワード×18ビット容量で、係数との乗算の部分積を生成するＲＯＭ。90～93は26ビット全加算器。94～97は26ビットのデータロード機能付き右方シフター。98,99は33ビット全加算器。100,101は33ビットレジスタ、102は33ビット出力信号ν(2k')、103は33ビット出力信号ν(2k'+1)である。第１図と第２図と第３図を用いて、８×１の一次元ＤＣＴ処理の動作について説明する。本発明においては、Ｍ＞８ビット長の１画素データをＬビット長のデータに分割して、処理を実行する。例えばＭビット長のデータをＪ個のＬビット長データに分割すると、式(1-5)は次のように変形できる。

$$\nu(k)=\sum_{l=0}^{L-1}2^{l}\left(\sum_{n=0}^{N-1}a_l(n)\alpha(k)\cos\left[\frac{2\pi(2n+1)k}{2N}\right]\right)$$

$$+\sum_{l=0}^{L-1}2^{l}\left(\sum_{n=0}^{N-1}a_l(n)\alpha(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right]\right)$$

$$+\cdots\cdots$$

$$+\sum_{l=L(J-1)}^{M-1}2^{l}\left(\sum_{n=0}^{N-1}a_l(n)\alpha(k)\cos\left[\frac{\pi(2n+1)k}{2N}\right]\right) \qquad (1-12)$$

上式はＪ個の部分項の和によって成り立ち、各部分項はＬ回の加算により実行されることを意味している。Ｌ回の加算時間とＪ個の項を加算する時間の総和が、Ｎ個のデータのサンプリング時間よりも短ければ、目的とする高速処理が実現出来る。一例として、N=8、J=2の場合を考える。この時以下の式を満足するＭビット長のデータまで高速処理が可能である。

$8 \geq \mathrm{trune}(M/2+0.5)+1$；trune(・)切り捨て(1-13)

故にＭ≦14となる。また、ＲＯＭ容量削減のため、従来例と同様に、式(1-12)に対し式(1-10),(1-11)を適用すると式(1-14),(1-15)が得られる。

k-2k',0≦K'≦3の時

$$\nu(2k')=\sum_{l=0}^{8}2^{l}\left(\sum_{n=0}^{8}(a_l(n)+a_l(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right)$$

$$+2^{7}\sum_{l=0}^{8}2^{l}\left(\sum_{n=0}^{8}(a_{l+7}(n)+a_{l+7}(7-n))\alpha(2k')\cos\left[\frac{2\pi(2n+1)k'}{8}\right]\right),\ 0\leq K'\leq 3 \qquad (1-14)$$

k-2k'+1,0≦K'≦3の時

$$\nu(2k'+1)=\sum_{l=0}^{8}2^{l}\left(\sum_{n=0}^{8}(a_l(n)-a_l(7-n))\alpha(2k'+1)\cos\left[\frac{\pi(2n+1)(2k'+1)}{8}\right]\right)$$

$$+2^{7}\sum_{l=0}^{8}2^{l}\left(\sum_{n=0}^{8}(a_{l+7}(n)-a_{l+7}(7-n))\alpha(2k')\cos\left[\frac{\pi(2n+1)(2k'+1)}{8}\right]\right)\ 0\leq K'\leq 3 \qquad (1-15)$$

Ｍについて１回の加算が増加するが、パイプライン構成を適用することにより、Ｍ-14ビットの精度で８画素のサンプリングクロックの期間で８×１の一次元のＤＣＴ処理を実現することができる。第１図において、８×１の一次元ＤＣＴ処理の動作を説明する。14ビットの画像信号入力u(j)2は８画素のサブセットに対してＤＣＴ処理を施されるため、14ビットレジスタ３～10に、それぞれ{u(n),n=mod(j)・,0≦n≦7)と分割されて保持される。14ビットレジスタ３～10では、この８個のサブセットデータ列{u(n),0≦n≦7)が完全に更新されるまで、１回のデータサンプリングに対して１回のシフト動作を行ない、データを順次送っていく。つまり、８回のデータサンプリング毎に、新しいサブセットデータが、14ビットレジスタ３～10にu(7)、・・・・、u(0)としてセットされる。次に、このデータは、14ビットの信号線11～18を介して、それぞれビットシリアル演算部19～22に供給される。このビットシリアル演算部19～22における処理を、第２図を用いて説明する。14ビットの画像入力65～66は、第１図の14ビットレジスタ３～10のいずれかからのデータで、２の補数表現を用いて現わすと、$u(n)=\sum_{i=0}^{13}a_i(n)2^i$($a_{13}(n)\in[0,-1]$,$a_i(n)\in[0,1]$,$0\leq i\leq 12$,$0\leq n\leq 3$)と、$u(7-n)=\sum_{i=0}^{13}a_i(7-n)2$

'(a_{i+8}(7-n)∈[0,-1], a_i(7-n)∈[0,1], 0≦i≦12, 0≦n≦3)である。これらのデータが、上位7ビットと下位7ビットが独立した14ビットのデータロード機能付き右方シフター67,68に入力され、それぞれ、$u(n)=\sum_{i=0}^{12} a_i(n)2^i+2^7\sum_{i=0}^{6} a_{i+8}(n)2^i$、$u(7-n)=\sum_{i=0}^{6} a_i(7-n)2^i+2^7\sum_{i=0}^{6} a_{i+8}(7-n)2^i$として上位7ビットと下位7ビットが分離した形で処理され、1クロック期間毎に1回のＬＳＢ側への右方シフトが実行される。データロード機能付き右方シフター67,68より出力される信号は、u(n)およびu(7-n)の上位7ビットと下位7ビットに関して2'桁の各1ビットの値で、a_i(n)とa_{i+8}(n)とa_i(7-n)とa_{i+8}(7-n)である。これらの信号により、1ビット全加算器69,70と1ビット全減算器71,72において、式(1-14)、(1-15)の右辺の(a_{i+8}(n)+a_{i+8}(7-n))、(a_i(n)+a_i(7-n))、((a_{i+8}(n)-a_{i+8}(7-n))、(a_i(n)-a_i(7-n))の演算を実行する。これらの演算により発生するキャリーおよびボローは1ビットラッチ73～76に保持され、1クロック後の演算に用いられるために、元の1ビット全加算器69,70と1ビット全減算器71,72に再帰的に入力される。1ビット全加算器69,70の演算結果は、1ビットデータ線77,78に各々出力され、1ビット全減算器71,72の演算結果は、1ビットデータ線79,80に各々出力される。

第2図で説明したのと同様に、ビットシリアル演算部19,22では、式(1-4)、(1-5)の右辺の(a_{i+8}(n)+a_{i+8}(7-n))、(a_i(n)+a_i(7-n))、((a_{i+8}(n)-a_{i+8}(7-n))、(a_i(n)-a_i(7-n))の演算が実行され、ビットシリアル演算部19ではu(0)とu(7)について、ビットシリアル演算部20ではu(1)とu(6)について、ビットシリアル演算部21ではu(2)とu(5)について、ビットシリアル演算部22ではu(3)とu(4)について、この演算を実行する。この結果、各ビットシリアル演算部19～22より出力される4ビットデータ線39～42は、4ビットデータ線39が{(a_{i+8}(n)+a_{i+8}(7-n))、n=0,1,2,3}を示し、4ビットデータ線40が{(a_i(n)+a_i(7-n))、n=0,1,2,3}を示し、4ビットデータ線41が{(a_{i+8}(n)-a_{i+8}(7-n))、n=0,1,2,3}を示し、4ビットデータ線42が{(a_i(n)-a_i(7-n))、n=0,1,2,3}をそれぞれ示している。これらの4ビットの信号の意味を、もう少し詳しく説明するために、式(1-14)、(1-15)に戻って説明する。式(1-14)および式(1-15)のnに関する和の部分を展開すると、次式のように表現することが出来る。

k=2k', 0≦K'≦3の時

$$\nu(2k')=\sum_{i=0}^{6} 2^i\{(a_i(0)+a_i(7))\alpha(2k')\cos\left[\frac{2\pi k'}{8}\right]$$
$$+(a_i(1)+a_i(6))\alpha(2k')\cos\left[\frac{6\pi k'}{8}\right]$$
$$+(a_i(2)+a_i(5))\alpha(2k')\cos\left[\frac{10\pi k'}{8}\right]$$
$$+(a_i(3)+a_i(4))\alpha(2k')\cos\left[\frac{14\pi k'}{8}\right]\}$$
$$+2^7\sum_{i=0}^{6} 2^i\{(a_{i+8}(0)+a_{i+8}(7))\alpha(2k')\cos\left[\frac{2\pi k'}{8}\right]\}$$
$$+(a_{i+8}(1)+a_{i+8}(6))\alpha(2k')\cos\left[\frac{6\pi k'}{8}\right]$$
$$+(a_{i+8}(2)+a_{i+8}(5))\alpha(2k')\cos\left[\frac{10\pi k'}{8}\right]$$
$$+(a_{i+8}(3)+a_{i+8}(4))\alpha(2k')\cos\left[\frac{14\pi k'}{8}\right]\}$$
$$,0\leqq k'\leqq 3 \qquad (1\text{-}16)$$

k=2k'+1, 0≦K'≦3の時

$$\nu(2k'+1)=\sum_{i=0}^{6} 2^i\{(a_i(0)-a_i(7))\alpha(2k'+1)\cos\left[\frac{\pi(2k'+1)}{8}\right]$$
$$+(a_i(1)-a_i(6))\alpha(2k'+1)\cos\left[\frac{3\pi(2k'+1)}{8}\right]$$
$$+(a_i(2)-a_i(5))\alpha(2k'+1)\cos\left[\frac{5\pi(2k'+1)}{8}\right]$$
$$+(a_i(3)-a_i(4))\alpha(2k'+1)\cos\left[\frac{7\pi(2k'+1)}{8}\right]\}$$
$$+2^7\sum_{i=0}^{6} 2^i\{(a_{i+8}(0)-a_{i+8}(7))\alpha(2k'+1)\cos\left[\frac{\pi(2k'+1)}{8}\right]$$
$$+(a_{i+8}(1)-a_{i+8}(6))\alpha(2k'+1)\cos\left[\frac{3\pi(2k'+1)}{8}\right]$$
$$+(a_{i+8}(2)-a_{i+8}(5))\alpha(2k'+1)\cos\left[\frac{5\pi(2k'+1)}{8}\right]$$
$$+(a_{i+8}(3)-a_{i+8}(4))\alpha(2k'+1)\cos\left[\frac{7\pi(2k'+1)}{8}\right]$$
$$,0\leqq k'\leqq 3 \qquad (1\text{-}17)$$

このように、上式(1-16)における各2'桁に関する演算は、K'を固定すれば{(a_{i+8}(n)+a_{i+8}(7-n))、n=0,1,2,3}の4ビットのデータと{(a_i(n)+a_i(7-n))、n=0,1,2,3}の4ビットのデータによって

一意的に決定することが出来る。又、式(1-17)についても同様のことが成立する。故にこれらの4ビット信号をアドレス情報とし、そのアドレス情報に従い$\{\sum_{n=0}^{3}(a_{i}(n)+a_{i}(7-n))\alpha(2k')\cos[\frac{2\pi(2n+1)k'}{8}]\}$を出力するようにROM化することは容易である。このように、4ビットデータ線39の4ビットデータは、式(1-14)における2'桁での$\{\sum_{n=0}^{3}a_{i+2}(n)+a_{i+2}(7-n))\alpha(2k')\cos[\frac{2\pi(2n+1)k'}{8}]\}$を求めるアドレス情報として用られ、ROMと加算器による係数乗算部43～46に入力される。同様に、4ビットデータ線40の4ビットデータは、式(1-14)における2'桁での$\{\sum_{n=0}^{3}(a_{i}(n)+a_{i}(7-n))\alpha((2k')\cos[\frac{2\pi(2n+1)k'}{8}]\}$を求めるアドレス情報として用いられ、ROMと加算器による係数乗算部43～46に入力される。同様に、4ビットデータ線41の4ビットデータは、式(1-14)における2'桁での$\{\sum_{n=0}^{3}(a_{i}(n)+a_{i}(7-n))\alpha(2k')\cos[\frac{2\pi(2n+1)k'}{8}]\}$を求めるアドレス情報で、4ビットデータ線40を介して入力される。同様に、4ビット信号84は、式(1-15)における2''桁での$\{\sum_{n=0}^{3}(a_{i+2}(n)-a_{i+2}(7-n))\alpha(2k')\cos[\frac{\pi(2n+1)(2k'+1)}{8}]\}$を求めるアドレス情報で、4ビットデータ線41を介して入力される。同様に、4ビット信号85は、式(1-15)における2'桁での$\{\sum_{n=0}^{3}(a_{i+2}(n)-a_{i+2}(7-n))\alpha((2k')\cos[\frac{\pi(2n+1)(2k'+1)}{8}]\}$を求めるアドレス情報で、4ビットデータ線42を介して入力される。次に、16ワード×18ビット容量のROM86では、4ビット信号82をアドレス情報として受け$\{\sum_{n=0}^{3}(a_{i+2}(n)+a_{i+2}(7-n))\alpha(2k')\cos[\frac{2\pi(2n+1)k'}{8}]\}$値を18ビットのデータとして出力する。同様に、(1-15)における2''桁での$\{\sum_{n=0}^{3}(a_{i+2}(n)-a_{i+2}(7-n))\alpha(2k')\cos[\frac{\pi(2n+1)(2k'+1)}{8}]\}$を求めるドレス情報として用いられ、ROMと加算器による係数乗算部43～46に入力される。同様に、4ビットデータ線42の4ビットデータは、式(1-15)における2'桁での$\{\sum_{n=0}^{3}(a_{i}(n)+a_{i}(7-n))\alpha(2k')(2k')\cos[\frac{\pi(2n+1)(2k'+1)}{8}]\}$を求めるアドレス情報として用いられ、ROMと加算器による係数乗算部43～46に入力される。次に、ROMと加算器による係数乗算部43～46の中での処理について、第3図を用いて説明する。第3図において、4ビット信号82は、式(1-14)における2''桁での$\{\sum_{n=0}^{3}(a_{i+2}(n)+a_{i+2}(7-n))\alpha(2k')\cos[\frac{2\pi(2n+1)k'}{8}]\}$を求めるアドレス情報で、4ビットデータ線39を介して入力される。同様に、4ビット信号線83は、16ワード×18ビット容量のROM87では、4ビット信号83をアドレス情報として受け$\{\sum_{n=0}^{3}(a_{i}(n)+a_{i}(7-n))\alpha(2k')\cos[\frac{2\pi(2n+1)k'}{8}]\}$の値を18ビットのデータとして出力する。同様に、16ワード×18ビット容量のROM88では、4ビット信号84をアドレス情報として受け$\{\sum_{n=0}^{3}(a_{i+2}(n)-a_{i+2}(7-n))\alpha(2k')\cos[\frac{\pi(2n+1)(2k'+1)}{8}]\}$の値を18ビットのデータとして出力する。同様に、16ワード×18ビット容量のROM89では、4ビット信号85をアドレス情報として受け$\{\sum_{n=0}^{3}(a_{i}(n)-a_{i}(7-n))\alpha(2k')\cos[\frac{\pi(2n+1)(2k'+1)}{8}]\}$の値を18ビットのデータとし出力する。次に26ビット全加算器90～93と、26ビットのデータロード機能付き右方シフター94～97は、4組の26ビット累積加算器として働き、前記ROM86～89からの18ビットの出力データは、26ビット全加算器90～93の一方の入力のMSB側18ビットに入力される。26ビット全加算器90～93で

の加算結果は、それぞれ26ビットのデータロード機能付き右方シフター94～97でＬＳＢ側に（右方に）１ビットシフトされ、次のクロック期間で、前記ＲＯＭ86～89の出力と加算される。但し、この動作で、i-0の時には、26ビットのデータロード機能付きシフター94～97から26ビット全加算器90～93に入力されるデータは"0"に初期化される。この操作により、８回のクロック期間で、式(1-14)、(1-15)のそれぞれ４つの項が算出される。33ビット全加算器98～99では、26ビットシフター94～97の出力を加算する。ここで、26ビットシフター94と96の出力は加算時に2'で桁補正が行なわれ、式(1-14)、(1-15)のν(2k')、(2k'+1)の値を算出する。そして、33ビットレジスター100、101に、その演算結果をセットする。33ビットレジスター100、101は次の８クロックの期間、新しいサブセットに対してν(2k')、(2k'+1)の値が算出されるまで、現在の値を保持する。ここで第１図に戻って、説明を続ける。第３図における前記33ビットレジスター100、101からのデータ102、103は、第１図の47～54に対応し、他の３つのブロックの信号の出力信号と合わせて、ＤＣＴ処理された信号列{ν(k), 0≦k≦7}となる。この33ビット出力信号列{ν(k), 0≦k≦7}がそれぞれトライステートドライバー55～62により、時系列化されて出力端子63より出力される。第４図は本発明の一実施例によるアダプティブＤＣＴ処理装置の概略構成を示すものである。104は制御信号入力端子、105はデータストローブ信号入力端子、106は14ビットの画像信号入力端子、107は14ビットの参照画像信号入力端子、108は差分器、109はクリッピング回路、110は８×１の一次元のＤＣＴ処理回路111に対するタイミング信号生成回路、112はクリッピング・丸め込み処理回路、113は128ワード×16ビットのデュアルポートメモリ114への書き込み制御回路、115はデュアルポートメモリ114からの読み出し制御回路、116は８×１の一次元のＤＣＴ処理回路117に対するタイミング信号生成回路、118はクリッピング・丸め込み処理回路、119は14ビットの画像出力端子である。第４図は第１図の８×１のＤＣＴ処理回路ブロックを利用した８×８のアダプティブＤＣＴ処理装置の一例である。制御信号104によりアダプティブ処理を行なう場合は、差分器108において、14ビット画像信号入力106と、14ビット参照画像信号入力107の差分をとる。その結果の信号が前提とされている最大・最小のしきい値を越える場合は、クリッピング回路109でクリップされ、８×１の一次元のＤＣＴ処理回路111に入力される。クリップを行なわない場合は、差分器108からの信号がスルーされ、８×１の一次元のＤＣＴ処理回路111に入力され、８×１のＤＣＴ処理が施される。８×１の一次元のＤＣＴ処理回路111における処理タイミングは、データストローブ信号入力端子105より入力される14ビット画像信号入力106から入力される一組64個のデジタル画像信号の先頭の信号を示すストローブ信号をトリガーとして、タイミング信号生成回路110により制御される。次に、クリッピング・丸め込み処理回路112では、８×１の一次元のＤＣＴ処理回路111からの処理出力に対しクリッピング・丸め込み処理を行ない、その結果を128ワード×16ビットのデュアルポートメモリ114に入力する。128ワード×16ビットのデュアルポートメモリ114の書き込み、読みだしは、書き込み制御回路113、読みだし制御回路115により制御される。次に、８×１の一次元のＤＣＴ処理回路117では、128ワード×16ビットのデュアルポートメモリ114からの入力信号をＤＣＴ処理し、ここでの処理タイミングはタイミング信号生成回路116により制御される。８×１の一次元のＤＣＴ処理回路117からの出力データは、クリッピング・丸め込み処理回路118を通じて、14ビットの画像出力端子119に出力され、二次元の８×８のＤＣＴ処理が完結する。

なお本実施例では１画素データが14ビット長の時、７ビット長の信号に２分割したが、Ｍ＞Ｎを満たすＭビット長をＬビット長の信号に分割しても（ただしＬ１ビット長の信号に分割する場合を除く）同様の効果を有する。

発明の効果

以上、説明したごとく本発明によれば、Ｍビット長をＬ＜Ｎを満足するＬビット長に分割し、Ｌ

ビット長で部分積の演算を並列的に実行し、最後にそれらの中間結果の加算を実行する方式により、N＝8，J＝2の時に、8つのサンプリングクロックの期間で8×1の一次元のＤＣＴ処理を実現することができ、かつ内部演算精度をＭ＝14ビットの精度まで乗算器を用いずに確保することができ、その実用的効果は大きい。

4. 図面の簡単な説明

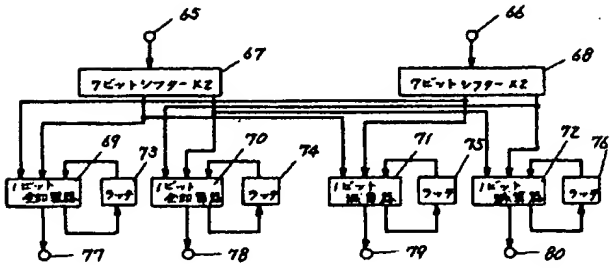
第1図は本発明の一実施例における8×1の一次元のＤＣＴ処理回路のブロック図、第2図はビットシリアル演算部の回路構成図、第3図はＲＯＭと加算器による係数乗算部の回路構成図、第4図は本発明の一実施例によるアダプティブＤＣＴ処理回路の概略構成図である。

2・・・・画像信号入力、3～10・・・・データレジスタ、19～22・・・・ビットシリアル演算部、43～46・・・・ＲＯＭと加算器による係数乗算部、111，117・・・・8×1の一次元ＤＣＴ処理回路、114・・・・デュアルポートメモリ。

代理人の氏名　弁理士　要野重孝　ほか1名

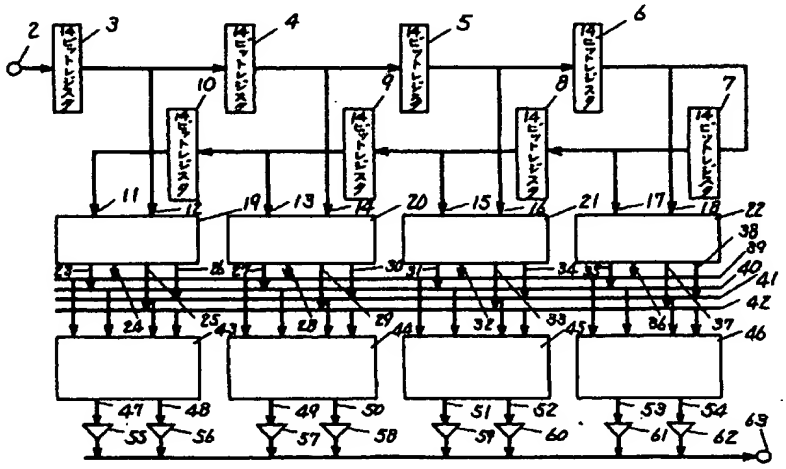
67，68 … 14ビットのデータロード機能付き右方シフター
69，70 … 1ビット全加算器
71，72 … 1ビット全減算器
73，76 … 1ビットデータラッチ

第2図

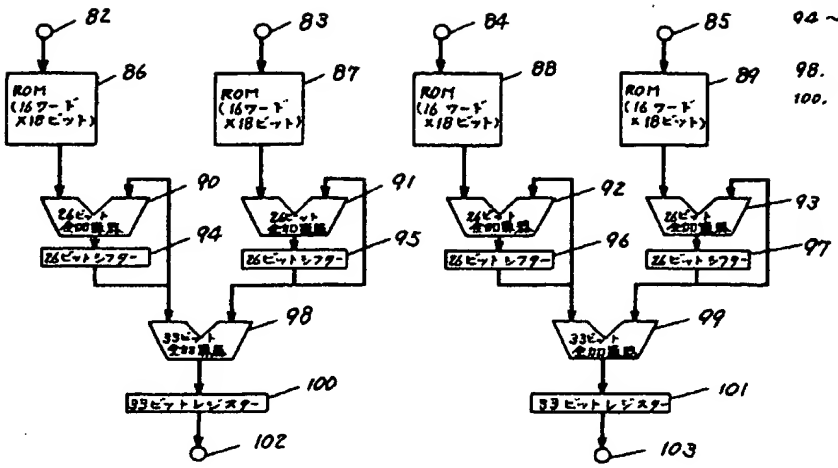


2 … 14ビットの画像信号入力
3～10 … 14ビットのデータレジスタ
11～18 … 14ビットの画像信号
19～22 … ビットシリアル演算部
43～46 … ＲＯＭと加算器による係数乗算器
55～62 … トライステートドライバ

第1図



第 3 図



86～89 … ROM
90～93 … 26ビット全加算器
94～97 … 26ビットのデータ機能付き右方シフター
98, 99 … 33ビット全加算器
100, 101 … 33ビットレジスタ

108 … 差分器
109 … クリッピング回路
110 … タイミング信号生成回路
111, 117 … 8×1の一次元DCT処理回路
112, 118 … クリッピング丸め込み処理回路
113 … 書き込み制御回路
114 … デュアルポートメモリ
115 … 読み出し制御回路

第 4 図